**Hands-on .NetCore 3.0**

**Read A Rest Api From Console**

Table of Contents

[Scaffolding Console App Basics 2](#_Toc23634326)

[ReadARestApiFrom Console 2](#_Toc23634327)

[Creating the console app 2](#_Toc23634328)

[Installing NuGet Extension 2](#_Toc23634329)

[Installing NEWTONSOFT JSON 3](#_Toc23634330)

[Free API Service for Testing 3](#_Toc23634331)

[Getting the data 4](#_Toc23634332)

[Serializing the JSON to object 5](#_Toc23634333)

[JSON => Object Representation 5](#_Toc23634334)

[Geostationary Class 6](#_Toc23634335)

[Address Class 6](#_Toc23634336)

[Company Class 6](#_Toc23634337)

[User Class 7](#_Toc23634338)

[Deserializing JSON to User 7](#_Toc23634339)

[Running the Code 8](#_Toc23634340)

# Scaffolding Console App Basics

We have already run the commands multiple times, here is the info

* dotnet new console
* dotnet restore [pulls in the dependencies needed by the application]
* dotnet run [compiles and run the application]
* dotnet build [compiles the application]
* dotnet publish [packages up the files for reuse]

Take a look at this document for more details:

<https://itplate.blogspot.com/2019/11/scaffolding-applications-with-net-cli.html>

# ReadARestApiFrom Console

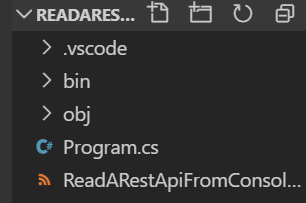
## Creating the console app

Run command **dotnet new console -o ReadARestApiFromConsole**

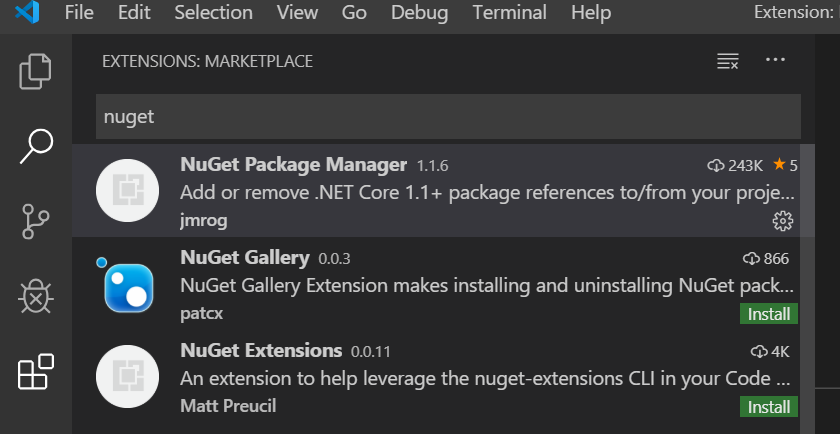
It has done the restore for us as well

Open the app with VS Code by

1. **cd ReadARestApiFromConsole**
2. and then typing **code .** [code space dot]

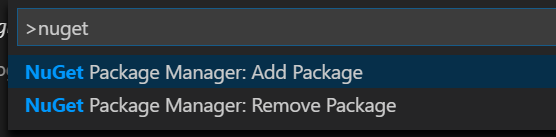


## Installing NuGet Extension

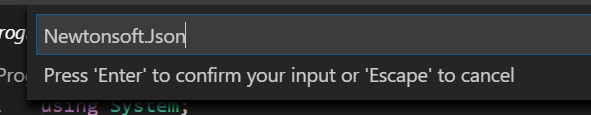


## Installing NEWTONSOFT JSON

To add newton soft press Ctrl+Shift+P and type >nuget and then select Add Package and hit enter

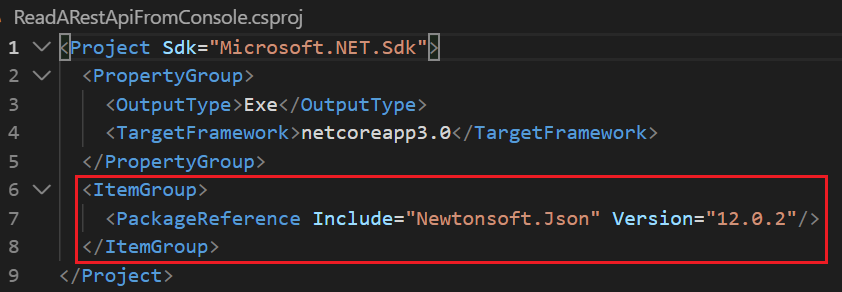


Then type NewtonSoft.Json and hit enter



Confirm NewtonSoft.Json and then confirm the latest version and hit enter.

Check the csproj file



## Free API Service for Testing

We’ll be connection to free service [https://jsonplaceholder.typicode.com](https://jsonplaceholder.typicode.com/) to test an API.

Full address is that we’ll test is following and will be getting the user with id 1.

<https://jsonplaceholder.typicode.com/users/1>

Here is the JSON returned

{

"id": 1,

"name": "Leanne Graham",

"username": "Bret",

"email": "Sincere@april.biz",

"address": {

"street": "Kulas Light",

"suite": "Apt. 556",

"city": "Gwenborough",

"zipcode": "92998-3874",

"geo": {

"lat": "-37.3159",

"lng": "81.1496"

}

},

"phone": "1-770-736-8031 x56442",

"website": "hildegard.org",

"company": {

"name": "Romaguera-Crona",

"catchPhrase": "Multi-layered client-server neural-net",

"bs": "harness real-time e-markets"

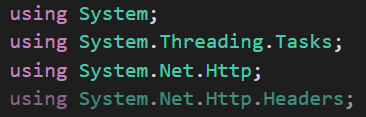
}

}

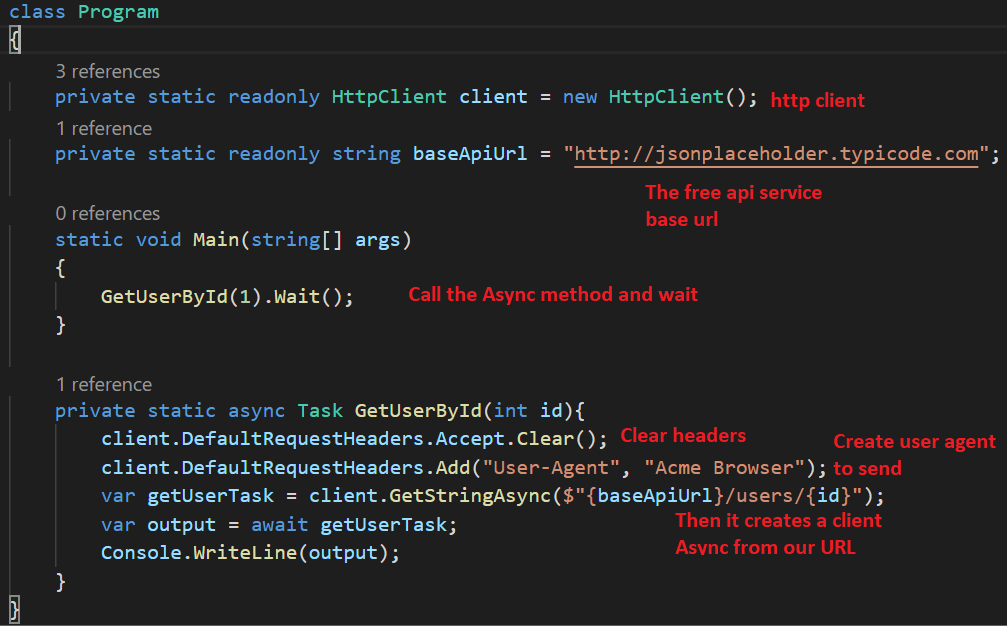
## Getting the data

We’ll be using the http client to connect to the above service and get data. It will be Async call.

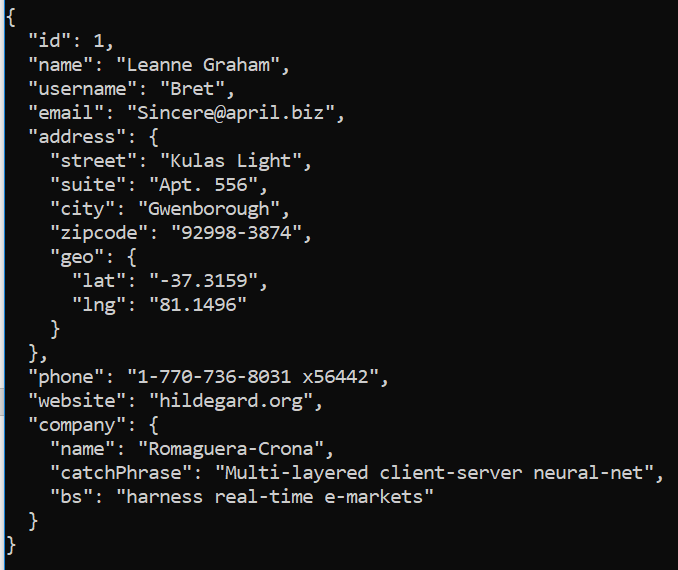
Make sure to add the following using statement



And then write the following code

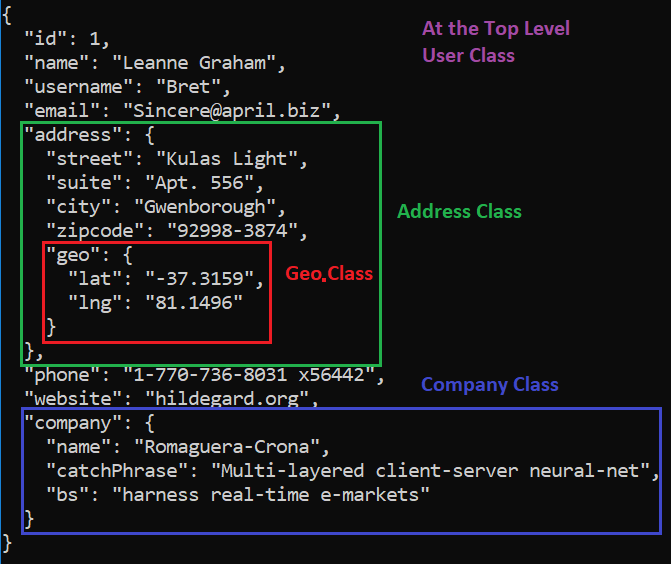


And finally perform **dotnet run** to look at the results



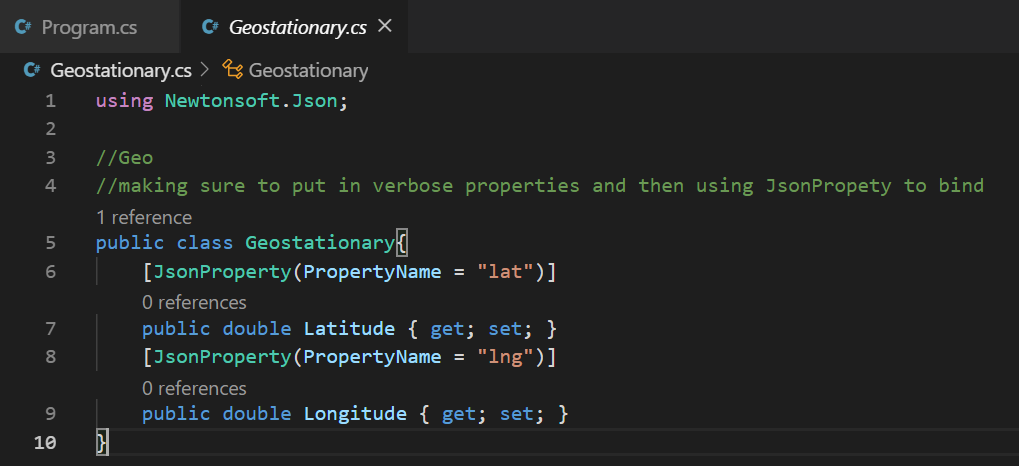
## Serializing the JSON to object

### JSON => Object Representation



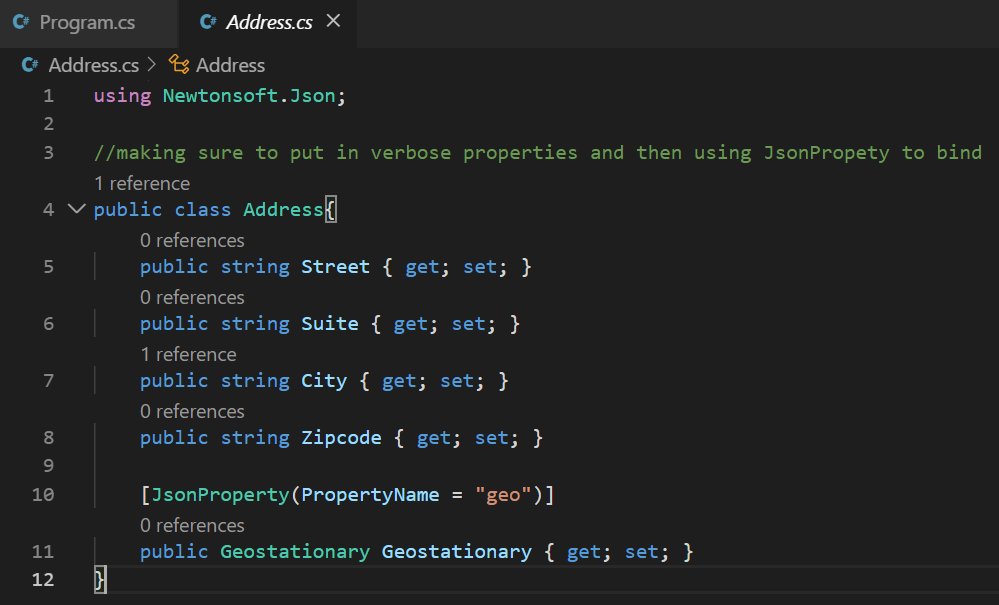
### Geostationary Class

Create a Geo Class with two properties. Right click inside explorer and click New File and name it Geostationary.cs



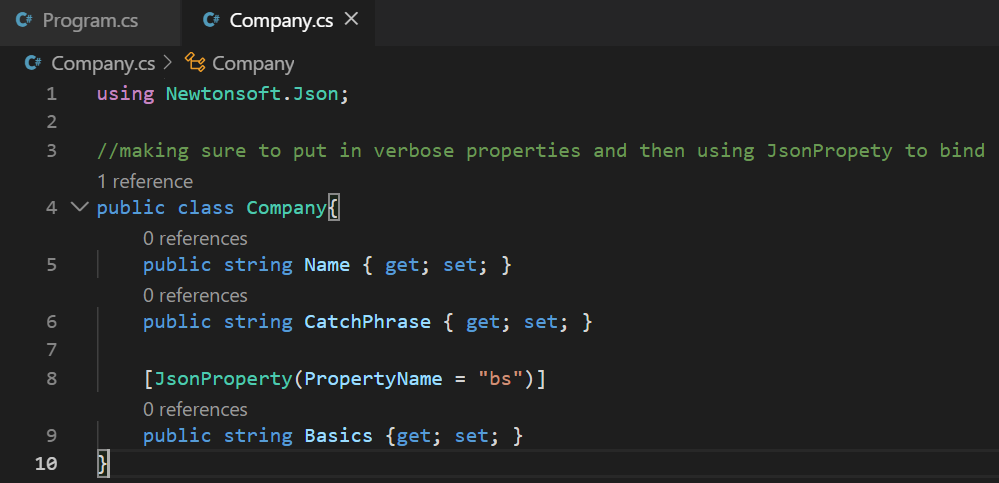
### Address Class

Right click in the explorer and click New File with Name Address.cs. It will have 5 properties.



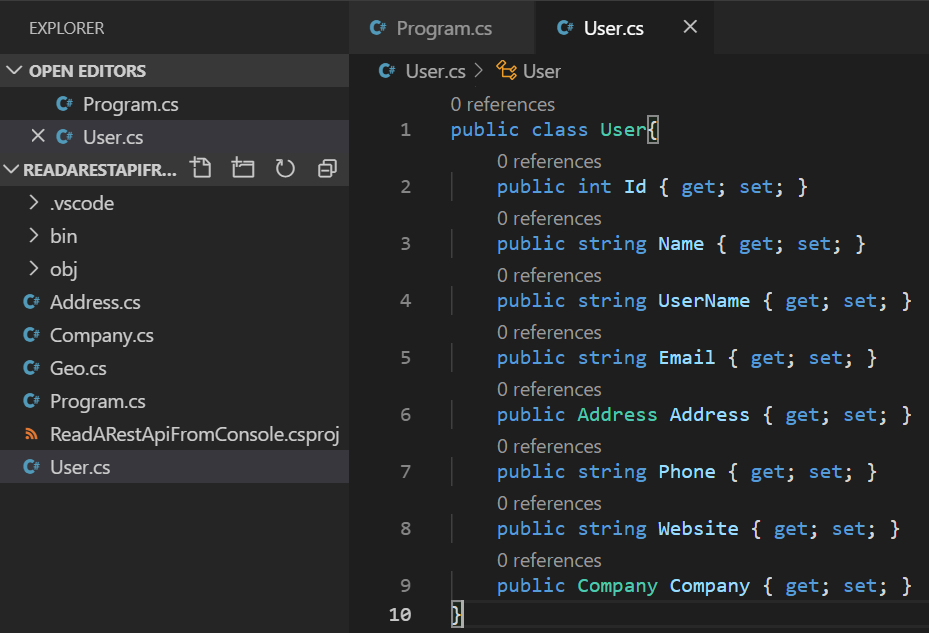
### Company Class

Right click in the explorer and click New File with Name Company.cs. It will have 3 properties.



### User Class

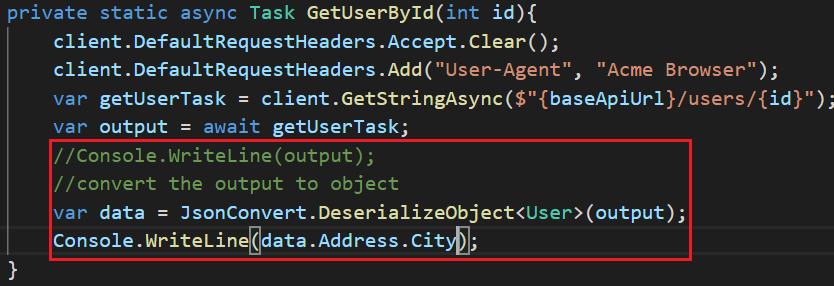
Right click in the explorer and click New File with Name User.cs. It will have 8 properties.



## Deserializing JSON to User

We’ll be using the NewtonSoft.Json for our purposes. Make sure to add **using Newtonsoft.Json;**

And then make the code changes as below



## Running the Code

Put a break point in the code and **press F5**

Then **press F10** to step through the code and look at the data variable for its contents. It should have the data.

